WO 02/11120
PCT

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
21 February 2002 (21.02.2002)

PCT

(10) International Publication Number
WO 02/15014 A1

(51) International Patent Classification[7]: G06F 13/00

(21) International Application Number: PCT/US01/23948

(22) International Filing Date: 31 July 2001 (31.07.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/637,803 11 August 2000 (11.08.2000) US

(71) Applicant: IP DYNAMICS, INC. [US/US]; Fifth floor, 1901 S. Bascom Avenue, Campbell, CA 95008 (US).

(72) Inventors: WOOTTON, Bruce, C.; 1992 Barbara Drive, Palo Alto, CA 94303 (US). ALKHATIB, Hasan, S.; 15725 Apollo Heights Court, Saratoga, CA 95070-6361 (US).

(74) Agent: MAGEN, Burt; Vierra Magen Marcus Harmon & DeNiro LLP, 685 Market Street, Suite 540, San Francisco, CA 94105-4206 (US).

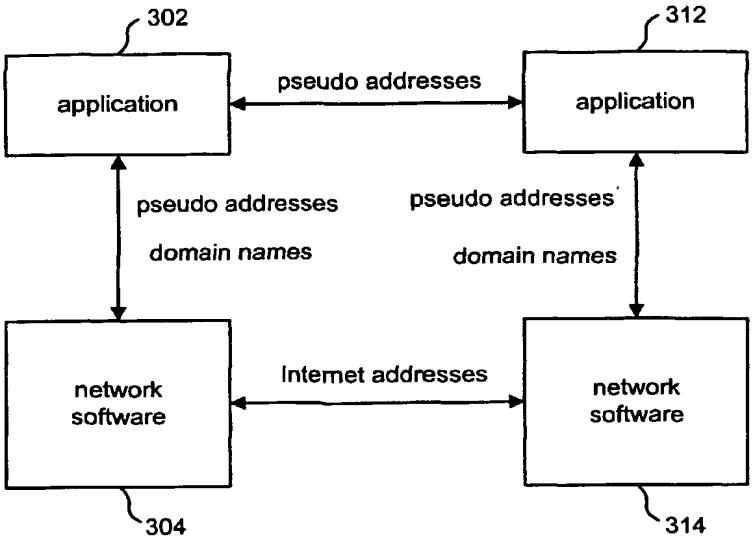
(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: PSEUDO ADDRESSING



(57) Abstract: The present invention provides for a system for communicating using pseudo addresses. Various embodiments of this system alleviate the diminishing IP address problem, allows for communication to continue after an entity (302, 312) has changed addresses and/or insulate the application software (302, 312) from the addressing formats of lower level protocols. The present invention also allows for communication to be initiated by a source entity (302) outside a private network that is directed to a destination entity (312) with a private address within the private network.

WO 02/15014 A1

PSEUDO ADDRESSING

CROSS REFERENCE TO RELATED APPLICATIONS

This application is related to the following Patents/Applications:

DOMAIN NAME ROUTING, Hasan S. Alkhatib, Serial No. 09/492,565, filed January 27, 2000, which is a continuation of U.S. Application 09/015,840, filed January 29, 1998;

IPNET GATEWAY, Hasan S. Alkhatib and Bruce C. Wootton, U.S. Application Serial No. 09/167,709, filed on October 6, 1998; and

COMMUNICATION USING TWO ADDRESSES FOR AN ENTITY, by Hasan S. Alkhatib and Fouad A. Tobagi, Attorney Docket No. TTCC-01003US1, filed the same day as the present application.

Each of the related Patents/Applications are incorporated herein by reference.

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention is directed to a system for communicating with entities on a network.

Description of the Related Art

Most machines on the Internet use TCP/IP (Transmission Control Protocol/Internet Protocol) to send data to other machines on the Internet. To transmit data from a source to a destination, the Internet Protocol (IP) uses an IP address. An IP address is four bytes long, which consists of a network number and a host number.

There are at least three different classes of networks currently in use: Class A, Class B and Class C. Each class has a different format for the combination of the network number and the host number in the IP addresses. Class A addresses include one byte to specify the network and three bytes to specify the host. The first bit of a

Class A address is a 0 to indicate Class A. Class B addresses use two bytes for the network address and two bytes for the host address. The first two bits of the Class B address are 10 to indicate Class B. The Class C address includes three bytes to specify the network and one byte for the host address. The first three bits of the Class C network address are 110 to indicate Class C. The formats described above allow for 126 Class A networks with 16 million hosts each; 16,382 Class B networks with up to 64K hosts each; and 4 million Class C networks with up to 256 hosts each.

When written out, IP addresses are specified as four numbers separated by dots (e.g. 198.68.70.1). Users and software applications rarely refer to hosts or other resources by their numerical IP address. Instead of using numbers, they use ASCII strings called domain names. A domain name is usually in the form of prefix.name_of_organization.top_level_domain. There are two types of top level domains: generic and countries. The generic domains are com (commercial), edu (educational institutions), gov (the U.S. Federal Government), int (international organizations), mil (the U.S. Armed Forces), net (network providers), and org (non-profit organizations). The country domains include one entry for each country. An example of a domain name is saturn.ttc.com. The term "saturn" is the prefix and may refer to a particular host in the network. The phrase "ttc" is the name of the organization and can be used to identify one or more networks to the outside world. The phrase "com" signifies that this address is in the commercial domain. The Internet uses a Domain Name System (DNS) to convert the domain name to an IP address.

The Internet Protocol has been in use for over two decades. It has worked extremely well, as demonstrated by the exponential growth of the Internet. Unfortunately, the Internet is rapidly becoming a victim of its own popularity: it is running out of addresses. Over 4 billion addresses exist, but the practice of organizing the address space into classes wastes millions of addresses. In particular, the problem is the Class B network. For most organizations, a Class A network, with 16 million addresses is too big, and a Class C network with 256 addresses is too small. A Class B network appears to be the right solution for most companies. In reality, however, a Class B address is far too large for most organizations. Many Class B networks have fewer than 50 hosts. A Class C network would have done the job, but many organizations that ask for Class B networks thought that one day they would outgrow the 8 bit host field.

One proposed solution to the depleting address problem is Classless Inter Domain Routing (CIDR). The basic idea behind CIDR is to allocate the remaining Class C networks in varied sized blocks. If a site needs 2,000 addresses, it is given a block of contiguous Class C networks, and not a full Class B network address. In addition to using blocks of contiguous Class C networks as units, the allocation rules for Class C addresses are also changed by partitioning the world into four zones. Each zone includes a predefined number of Class C networks. Although CIDR may buy a few more years time, IP addresses will still run out in the foreseeable future.

Another proposed solution is Network Address Translation (NAT). This concept includes predefining a number of Class A, Class B and Class C network addresses to be local addresses (also called private addresses). The remainder of the addresses are considered global addresses (also called public addresses). Global addresses are unique addresses that should only be used by one entity having access to the Internet. That is, no two entities on the Internet should have the same global address. Local addresses are not unique and are typically used for entities not having direct access to the Internet. Local addresses can be used by more than one organization or network. In the past, a local address could not be used to route on the Internet. Local addresses traditionally can only be used within a private network. NAT assumes that all of the machines on a private network will not need to access the Internet at all times. Therefore, there is no need for each machine to have a global address. A company can function with a small number of global addresses assigned to one or more gateway computers. The remainder of the machines on the private network will be assigned local addresses. When a particular machine on the private network using a local address attempts to initiate a communication to a machine outside of the private network (e.g. via the Internet), the gateway machine will intercept the communication, change the source machine's local address to a global address and set up a table for translation between global addresses and local addresses. The table can contain the destination address, port numbers, sequencing information, byte counts and internal flags for each connection associated with a host address. Inbound packets are compared against entries in the table and permitted through the gateway only if an appropriate connection exists to validate their passage. One problem with the NAT approach is that it only works for communication initiated by a host within the private network to a host on the Internet which has a global IP address.

The NAT approach specifically will not work if the communication is initiated by a host outside of the private network and is directed to a host with a local address in the private network.

A solution to the diminishing address problem that uses local and global addresses, and allows a communication to be initiated from outside the private network, needs a means for the application software to identify an entity with a local address. The application cannot use the local address because the local address does not uniquely identify the entity. The application cannot use the entity's global address because either the entity does not have a global address, the global address is temporary (and, therefore, not reliable), or the global address is not unique to the one entity.

Another solution that has been proposed is a new version of the Internet Protocol called IPv6 (Internet Protocol version 6, also known as IPng). IPv6 is not compatible with the existing Internet Protocol (IPv4). For example, IPv6 has a longer address than IPv4. Because IPv6 is not compatible with IPv4, almost all application software will need to be replaced with newer software that can use the IPv6 address. Such widespread updating of software is not practical, too expensive and not likely to be accepted by the millions of computer users.

Another technology that has ben effected by the IP address problem is mobile Internet access (including wireless access). Many mobile computing devices are now able to access the Internet, for example, laptop computers, cellular telephones, handheld computers (e.g. Palm Pilot), etc. There are not enough Internet addresses available to assign a static global Internet address to each mobile computing device. Thus, these devices are typically assigned temporary Internet addresses by an Internet Service Provider or other entity. Often, a mobile computing device's Internet address may change during a communication. For example, a cellular telephone may change its Internet address as it is moved to a different location. Currently, a communication between two entities cannot be maintained if one of the entities changes its Internet address.

SUMMARY OF THE INVENTION

The present invention, roughly described, provides for a system for communicating using pseudo addresses. Various embodiments of this system alleviate the diminishing IP address problem discussed above, allow for communication to

continue after an entity has changed addresses and/or insulate the application software from the addressing formats of lower level protocols. The present invention also allows for communication to be initiated by a source entity outside a private network that is directed to a destination entity with a local address within the private network.

One embodiment of the present invention includes a method for communicating. The method includes the step of receiving a request to communicate from a first application on a source entity. The request includes a first pseudo address used by the source entity to identify a destination entity. The first pseudo address is used to access a global address. A quantity of information is sent from the source toward the destination entity using the global address. One embodiment of the present invention further includes the steps of receiving the quantity of information at the destination entity, providing at least a subset of the quantity of information to an application on the destination entity and providing the pseudo address to the application on the destination entity.

One alternative embodiment further includes the steps of receiving the quantity of information at an intermediate entity, where the quantity of information includes a source address and a destination address. The destination address is a global address corresponding to the intermediate entity. The intermediate entity accesses a local address for the destination and sends the quantity of information to the destination using the local address.

The present invention can be accomplished using hardware, software, or a combination of both hardware and software. The software used for the present invention is stored on one or more processor readable storage media including a hard disk drive, CD-ROM, optical disk, floppy disk, RAM, ROM or other suitable storage device. In alternative embodiments, some or all of the software can be replaced by dedicated hardware including custom integrated circuits, gate arrays, FPGAs, PLDs, and special purpose computers.

These and other objects and advantages of the invention will appear more clearly from the following detailed description in which the preferred embodiment of the invention has been set forth in conjunction with the drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 depicts an IP packet.

Figure 2 shows the format of a header of an IP packet.

Figure 3 is a block diagram of multiple entities connected to the Internet.

Figure 4 is a block diagram of one embodiment of hardware suitable for use with the present invention.

Figure 5 is a block diagram illustrating the use of pseudo addresses.

Figure 6 is a flow chart describing a high level operation according to one embodiment of the present invention.

Figure 7 is a flow chart describing a process of domain name resolution according to a first embodiment of the present invention.

Figure 8 depicts a table entry.

Figure 9 is a flow chart describing a process for starting communication according to a first embodiment of the present invention.

Figure 10 is a block diagram illustrating encapsulated packets.

Figure 11 depicts the format of information stored in an options field.

Figure 12 is a flow chart of the process of communicating according to a first embodiment of the present invention

Figure 13 is a flow chart describing a process of domain name resolution according to second embodiment of the present invention.

Figure 14 is a flow chart describing a process for starting communication according to a second embodiment of the present invention.

Figure 15 is a flow chart of the process of communicating according to a second embodiment of the present invention.

DETAILED DESCRIPTION

The TCP/IP reference model for designing and building a network includes at least four layers: the Physical and Data Link Layer, the Network Layer, the Transport Layer, and the Application Layer. The physical layer portion of the Physical and Data Link Layer is concerned with transmitting raw bits over a communication channel. The design issues include ensuring that when one side sends a 1 bit it is received by the other side as a 1 bit, not as a 0

bit. Typical questions addressed are how many volts should be used to represent a 1 bit, how many volts to represent a 0 bit, how many microseconds a bit lasts, whether transmissions may proceed simultaneously in both directions, how the initial connection is established, how it is torn down when both sides are finished, and how many pins the network connector has. The data link portion of Physical and Data Link Layer takes the raw transmission facility and transforms it into a line that appears to be relatively free of transmission errors. It accomplishes this task by having the sender break the input data up into frames, transmit the frames and process the acknowledgment frames sent back by the receiver.

The Network Layer permits a host to inject packets into a network and have them travel independently to the destination. The protocol used for the Network Layer on the Internet is called the Internet Protocol (IP). The main function of the Network Layer is routing packets from a source entity to a destination entity. In most cases, packets will require multiple hops to make the journey. The Network Layer software uses one or more routing methods for deciding which output line an incoming packet should be transmitted on. There are many routing methods that are well known in the art that can be used in a network layer. For purposes of this patent, no specific routing method is required. Any suitable routing method known in the art will suffice.

The network entity, the process implementing the network layer, receives a segment from the transport layer process. The network entity appends a header to the segment to form a packet. The packet is sent to a router on a network (e.g. the Internet). Each router has a table listing IP addresses for a number of distant networks and IP addresses for hosts in the network closest to the router. When an IP packet arrives, its destination address is looked up in the routing table. If the packet is for a distant network, it is forwarded to the next router listed in the table. If the distant network is not present in the router's tables, the packet is forwarded to a default router with more extensive tables. If the packet is for a local host (e.g. on the router's Local Area Network (LAN)), it is sent directly to the destination.

Although every machine in the Internet has an IP address, these addresses alone cannot be used for sending packets because the data link layer does not understand Internet addresses. Most hosts are attached to a LAN by an interface board

that only understands LAN addresses. For example, every Ethernet board comes equipped with a 48 bit Ethernet address. Manufacturers of Ethernet boards request a block of addresses from a central authority to ensure that no two boards have the same address. The boards send and receive frames based on a 48 bit Ethernet address. For one entity to transmit data to another entity on the same LAN using an Ethernet address, the entity can use the Address Resolution Protocol (ARP). This protocol includes the sender broadcasting a packet onto the Ethernet asking who owns the particular IP address in question. That packet will arrive at every machine on the Ethernet and each machine will check its IP address. The machine that owns the particular IP address will respond with its Ethernet address. The sending machine now has the Ethernet address for sending data directly to the destination on the LAN. At this point, the Data Link Layer on the sender builds an Ethernet frame addressed to the destination, puts the packet into the payload field of the frame and dumps the frame onto the Ethernet. The Ethernet board on the destination receives the frame, recognizes it is a frame for itself, and extracts the IP packet from the frame.

The Transport Layer is designed to allow peer entities on the source and destination to carry on a "conversation." On the Internet, two end-to-end protocols are used. The first one, the Transmission Control Protocol (TCP), is a reliable connection-oriented protocol that allows a byte stream originating on one machine to be delivered without error to another machine on the Internet. It fragments the incoming byte stream into discrete packets and passes each one to the Network Layer. At the destination, the receiving TCP process reassembles the received packets into the output stream. TCP also handles flow control to make sure a fast sender cannot swamp a slow receiver with more packets than it can handle. The second protocol used in the Transport Layer on the Internet, User Datagram Protocol (UDP), is an unreliable connectionless protocol for applications that do not want TCP sequencing or flow control. UDP is used for one-shot, client server type requests-reply queries for applications in which prompt delivery is more important than accurate delivery. The Transport Layer is considered to be above the Network Layer to indicate that the Network Layer provides a service to the Transport Layer. Similarly, the Transport Layer is below the Application Layer to indicate that the Transport Layer provides a service to the Application Layer. The Application Layer contains the high level

protocols, for example, Telnet, File Transfer Protocol (FTP), Electronic Mail - Simple Mail Transfer Protocol (SMTP), and HyperText Transfer Protocol (HTTP).

Communication in the Internet generally works as follows. The Transport Layer breaks up a stream of data from the Application Layer into a number of segments. The Network Layer, using the Internet Protocol, transports the segments in one or more IP packets from source to destination, without regard to whether these machines or entities are on the same network. Each segment can be fragmented into small units as it is transported. When all of the fragments finally get to the destination machine, they are reassembled by the Network Layer into the original segment. This segment is then handed to the Transport Layer, which inserts it into the receiving process' (Application Layer) input stream.

Figure 1 depicts the structure of an IP packet 10. IP packet 10 consists of header 12 and payload 14. Payload 14 stores the data received from the Transport Layer. Figure 2 depicts the format of a header of an IP packet. The header is depicted to include six rows. Each row is 32 bits wide. The first five rows of the header comprise a 20 byte fixed portion of the header. The last row of the header provides a variable sized options field 22. Version field 24 keeps track of which version of the protocol the packet belongs to. The current version used on the Internet is version 4. IHL field 26 describes the length of the header in 32 bit words. Type field 28 indicates the type of service requested. Various combinations of reliability and speed are possible. Length field 30 includes the size of the packet, including both the header and the data. Identification field 32 is needed to allow the destination host to determine which segment the received fragment belongs to. All fragments of a segment contain the same identification value. Next comes three flags, which include an unused bit 33 and then two 1 bit fields 34 and 36. In one embodiment of the present invention, the unused bit 33 is used to indicate that the packet was created according to the present invention and is storing both a global address and a local address for the destination. DF field 34 stands for don't fragment. It is an order to the routers not to fragment the segment because the destination is incapable of putting the pieces back together again. MF field 36 stands for more fragments. All fragments except for the last one have this bit set. Fragment offset field 38 indicates where in the current segment this fragment belongs. Time to Live field 40 is used to limit packet lifetime. It is supposed to count time in seconds, allowing a maximum life time of 255 seconds. In practice, it may

count hops. The time is decremented on each hop by a router. When the time to live hits 0, the packet is discarded and a warning is sent back to the source using an Internet Control Messaging Protocol (ICMP) packet. This feature prevents packets from wandering around forever. Protocol Field 42 indicates which transport layer type is to receive the segment. TCP is one possibility, UDP is another. The present invention is not limited to any particular protocol. Checksum field 44 verifies the header. One method for implementing a checksum is to add up all 16 bit half words as they arrive and take the ones compliment of the result. Note that the checksum must be recomputed at each hop because the Time to Live field 40 changes. Source field 46 indicates the IP address for the source of the packet and destination field 48 indicates the IP address for the destination of the packet.

Options field 22 is a variable length field designed to hold other information. Currently, options used on the Internet indicate security, suggested routing path, previous routing path and time stamps. In one embodiment of the present invention, it is contemplated that the source and/or destination's local addresses are added to Options field 22. In some alternatives, the two local addresses can be encoded, compressed, encrypted or otherwise altered to provide more efficient use of storage space, security or compatibility.

In another embodiment, the local addresses of the source, destination or both are added to the end of the data portion of a packet as a trailer. In this case, Length field 30 needs to account for the extra bytes added at the end of the data field. Legacy routers can treat this trailer as an integral part of the data field and ignore it. In yet another embodiment, source field 46 and destination field 48 can be enlarged to 64 bits each so that they each can store a local address and a global address.

In another embodiment in the present invention, the local address and global address of an entity are both stored in a packet by utilizing encapsulation. That is, one IP packet is encapsulated inside the payload of another IP packet.

Figure 3 shows two networks connected to Internet 138. The first network includes a Router connected to private network 144 which is made up of a number of entities using local addresses. Figure 3 shows three entities 146, 148 and 150; however, more or less than three entities can also be used. Entity 146 is labeled as A, and has a global IP address of GIP_A.

Figure 3 also shows a Gateway connected to Internet 138 and to private network 162. The Gateway has a global address of GIP_G. Figure 4 shows part of network 162 including three entities 166, 168 and 170; however, more or less than three entities can be used. Entity 170 is labeled as B and has a local address of LIP_B. In the description below, examples will be made referring to the entities depicted in Fig. 4. Other configurations will also work with the present invention. The present invention allows entity A to initiate a communication with entity B by using both the global address for the Gateway (GIP_G) and the local address for B (LIP_B). Similarly, B can initiate communication with A utilizing the global address for A (GIP_A). For example purposes, it is assumed that A and B are computers. Alternatively, A and B can be other electronic devices that can communicate on the Internet or other network.

Figure 3 also shows entity X and entity Y connected to the Internet. These entities can be computers or other devices. In one example, entity X is a mobile device that uses wireless communication to access the Internet (or other network).

Figure 4 shows one example of a hardware architecture for computers used to implement the present invention. The hardware includes a processor 202, a memory 204, a mass storage device 206, a portable storage device 208, a first network interface 210, a second network interface 212 and I/O devices 214. The choice of processor is not critical as long as a suitable processor with sufficient speed is chosen. Memory 204 can be any conventional computer memory. Mass storage device 206 can include a hard drive, CD-ROM or any other mass storage device. Portable storage 208 can include a floppy disk drive or other portable storage device. If the computer is acting as a router, it includes two or more network interfaces. In other embodiments, the computer may include only one network interface. The network interface can include a network card for connecting to an Ethernet or other type of LAN. In addition, one or more of the network interfaces can include or be connected to a firewall. For a gateway, one of the network interfaces will typically be connected to the Internet and the other network interface will typically be connected to a LAN. However, a gateway can exist physically inside a network. I/O devices 214 can include one or more of the following: keyboard, mouse, monitor, display, printer etc. Software used to perform the methods of the present invention are likely to be stored in mass storage 206 (or any form of non-volatile memory), a portable storage media (e.g. floppy disk or tape) and/or, at some point, in memory 204. Various embodiments, versions, and

modification of the system of Fig. 4 can be used to implement a gateway, a router, other host, etc. The above described hardware architecture is just one suitable example depicted in a generalized and simplified form. The present invention could include dedicated hardware, a dedicated router with software to implement the invention or other software and/or hardware architectures that are suitable.

Figure 5 is a block diagram illustrating the use of pseudo addresses. A pseudo address in its most generic form is an identification of an entity that is different from the entity's actual address. In one embodiment, the pseudo address is a random address chosen to identify a particular entity. The randomly chosen address is not the entity's actual address. In one alternative, the pseudo address is in the format for IPv4 addresses. Depicted in Figure 5 are application 302 and network software 304, which are both running on a single computer (in one embodiment). Network software 304 pertains to software at the transport layer, network layer, and other layers. Application software 302 pertains to software at the application layer. Figure 5 also shows application 312 and network software 314, both of which are running on a single computer (in one embodiment). In one example, application 302 communicates with application 312. According to the current invention, both applications 302 and 312 use pseudo addresses to communicate with each other. However, in actuality, application 302 communicates with application 312 using network software 304 and network software 314. Application 302 identifies application 312 to network software 304 using a domain name and a pseudo address. Network software 304 communicates with network software 314 using Internet addresses (e.g. global addresses). In one example, network software 304 and network software 314 use IPv4 addresses. Other address formats can also be used, including IPv6. Application 312 identifies application 302 to network software 314 using a pseudo address and a domain name. Thus, from the point of view of the application software, the pseudo addresses are being used to identify the applications. Therefore, if the Internet addresses of the two computers (or other entities) change, applications 302 and 312 do not need to know about the change in the Internet addresses, including changes in format, changes of actual address, etc. because the pseudo addresses have not changed. Additionally, because the applications are using pseudo addresses, the applications do not need to be concerned with the format or change of format of the IP addresses. Thus, an IPv4 application can be made

to work with IPv6. Note that if application 302 asks network software 304 for its own IP address, then network 304 will respond with the pseudo address.

Figure 6 is a flow chart describing a high level operation of one embodiment of the present invention. In step 330, the entity wishing to initiate communication performs a domain name resolution. For purposes of a first example, assume that entity X of Figure 3 is attempting to initiate communication with entity Y. In step 332, the communication is started. In step 334, the two entities communicate with each other. In step 336, the communication ends.

Figure 7 is a flow chart describing one embodiment of the method for domain name resolution, corresponding to step 330 of Figure 6. In step 350, the application software running on entity X requests a domain name resolution. Typically, when an application seeks to establish communication with an entity on the Internet, the application is only in possession of the entity's domain name. The application makes a call to a resolver process, which converts the domain name to an IP address. Every domain, whether it is a single entity or a top level domain, has a set of resource records associated with it. For a single entity, the most common resource record is its IP address. When a resolver process gives a domain name to the domain name system, it gets back the resource records associated with that domain name.

A resource record has five fields: domain name, time to live, class, type and value. The time to live field gives an indication of how stable the record is. Information that is highly stable is assigned a large value such as the number of seconds in a day. The third field is the class. For the Internet the class is IN. The fourth field tells the type of resource record. One domain may have many resource records. There are at least eight types of resource records that are important to this discussion: SOA, A, MX, NS, CNAME, PTR, HINFO, and TXT. The value field for an SOA record provides the name of the primary source of information about the name server zone, e-mail address of its administrator, a unique serial number and various flags and time outs in the value field. The value field for an A record holds a 32 bit IP address for the host. The value field for the MX record holds the domain name of the entity willing to accept e-mail for that particular domain name. The NS record specifies name servers. The CNAME record allows aliases to be created in the value field. A PTR record just points to another name in the value field, which allows look up of an IP address for a particular domain name. The value field of the HINFO record

indicates the type of machine and operating system that the domain name corresponds to.

When an application requests resolution of a domain name, the network software for that entity will contact a DNS server in order to obtain the authoritative resource record that indicates the IP address for the host associated with the domain name. This resource record will be returned to the network software for the entity X in step 352. In step 354, the network software for entity X will chose a pseudo address. In one embodiment, the pseudo address is chosen randomly. In other embodiments, a list of pseudo addresses to choose from can be prestored. In the current example, the pseudo address is in the same format as typical IPv4 Internet addresses. In step 356, the pseudo address chosen in step 354, the IP address resolved in step 352 and the domain name from step 350 are all stored in a table. In step 358, the pseudo address is provided to the application software.

Figure 8 shows an example of an entry in the table mentioned in step 356. The entry includes four fields: local pseudo address 402, remote pseudo address 404, remote local IP address 406 and remote global IP address 408. Using the example that entity X is initiating a communication with entity Y, assume that the entry of Figure 8 is on a table stored on entity X. Thus, local pseudo address 402 is a pseudo address used by entity Y to identify entity X. Remote pseudo address 404 is a pseudo address used by entity X to identify entity Y. Remote local IP address 406 is the private network address of entity Y (if entity Y has a private IP address) and remote global IP address 408 is a global IP address associated with entity Y. Note that in some embodiments, a table may contain less than all four fields. In other embodiments, this information can be stored in data structures other than a table. The exact format of the data structure is not important to the present invention.

Figure 9 is a flow chart describing the process of starting communication, which is step 332 of Figure 6. In step 500, network software 304 of entity X receives a request from application 302 of entity X to communicate with application 312 of entity Y. The request includes the pseudo address that application 302 uses to identify entity Y. Alternatively, the request can include an identification of entity Y by domain name or other means. In step 502, network software 304 of entity X creates a packet. This packet includes the pseudo address that entity X uses to identify entity Y. In one embodiment, the packet created in step 502 is an IP packet. The source field 46 of the

IP packet is the global IP address for entity X. The destination field 48 of the IP packet is the global IP address for entity Y. When network software 304 receives the pseudo address from application 302 of entity X, the pseudo address (remote pseudo address 404) is used to access the table to determine the global IP address 408 for entity Y. There are many different options for inserting the pseudo address of entity Y into the packet. One implementation uses encapsulation. A second implementation uses the options field 22 of the IP packet.

Figure 10 illustrates encapsulation. That is, Figure 10 shows three packets 620, 622 and 624. Packet 620 includes header portion 640 and data (or payload) portion 642. Packet 622 includes header portion 644 and payload portion 646. Packet 624 includes header portion 650 and payload portion 652. Packet 624 is encapsulated within packet 622. For example, packet 624 is included in the data portion 646 of packet 622. Packet 622 is encapsulated within packet 620. For example, packet 622 is within the data portion 642 of packet 620. In one embodiment, packet 624 is actually a TCP segment, packet 620 is a first IP packet and packet 622 is a second IP packet. Packet 622 has a source address equal to the IP address of X and a destination address equal to the global address of Y. Packet 620 is used to route on the Internet from entity X to entity Y. Packet 622 has a source equal to the global IP address of entity X, however, the destination is equal to the pseudo address that entity X uses to identify entity Y. Packet 622 is not used to route on the Internet. Rather, entity Y will read packet 622 and use it to access the pseudo address stored therein. In another embodiment, packet 622 can be a new format, called a pseudo address format. This new format would likely have fields for pseudo addresses and IP addresses. Packet 620 would have a flag to indicate that it is encapsulating a pseudo address packet. One option is to include a flag in the protocol field 42 to indicate that the encapsulated packet is a pseudo addresses packet.

Instead of using encapsulation to add a pseudo address to a packet, the pseudo address can be added to the options field 22 of the packet. Figure 11 shows the format for adding the pseudo address to option field 22. Data in the option field typically has a format including three fields: option type 670, length 672 and option 674. An option type would be created to indicate that the option is a pseudo address. Length 672 would indicate the length of the three fields. Option 674 would store the actual pseudo

address. As discussed above, one embodiment includes the pseudo address being in IPv4 format.

Looking back at Figure 9, in step 504, the packet is set from entity X to entity Y, via the Internet (or other network). In step 506, entity Y receives the packet. In step 508, entity Y accesses the pseudo address within the packet. In step 510, the pseudo address is added to a table on entity Y. In one embodiment, the table on entity Y is the same format as depicted in Figure 8. The pseudo address added to the table in step 510 is the pseudo address that entity X uses to identify entity Y. Step 510 also includes adding the remote global IP address 408 of entity X to the table. In step 512, entity Y chooses a pseudo address for entity X. As described above, one method for choosing a pseudo address includes randomly choosing an appropriately formatted address. Other embodiments include using prestored addresses. In step 514, the chosen address from step 512 is added to the table. In step 516, the remote pseudo address is provided to application 312 and entity Y. In step 518, entity Y, at the request of application 312 (of entity Y) or otherwise, creates a packet. The source of the packet is the IP address for entity Y and the destination of the packet is the IP address for entity X. The packet also includes a pseudo address that Y uses to identify X. In step 520, the packet is sent from entity Y to entity X. In step 522, entity X receives the packet. In step 524, entity X access the pseudo address from the packet. This is the pseudo address that entity Y uses to identify entity X. In step 526, the pseudo address is entered into the table on entity Y in local pseudo address 402.

Figure 12 is a flow chart describing the process for communicating between entity X and entity Y, which is step 334 of Figure 6. In step 700, network software 304 of entity X receives data and the pseudo address from application 302 of entity X, as part of a request to send that data to entity Y. In step 702, network software 304 accesses the table using the pseudo address to identify the global IP address for sending information to entity Y. In step 704, it is determined whether there has been a change in the connection. If there has not been a change in the connection, then in step 706, a packet is created. A packet includes the IP addresses for entity X and entity Y, but does not include any pseudo addresses.

During a communication between two entities, it is possible that the connection changes. For example, one of the entities may change its IP address. For example, if one of the entities is a cellular telephone traveling between two distinct areas, the IP

address may change when entering the new area. Other scenarios for an IP address changing also apply, as well as other reasons for changes in connections. If there is a change in connection (on the part of entity X) during communication, then (at step 704) the method loops to step 710. In step 710, a packet is created with the IP addresses for entity X and entity Y. The packet created by entity X will also include the pseudo address that entity Y uses to identify entity X. After step 706 or step 710, the method loops to step 708, and the packet is sent to the destination (e.g. entity Y). In step 712, the packet is received at entity Y. When the packet is received at entity Y, the IP address of the source of the packet is used to access the table to determine the pseudo address that entity Y uses to access entity X. In step 714, entity Y determines whether the packet contains a pseudo address. If the packet does not contain a pseudo address, then the data is presented to the application in step 716. Additionally, in step 716, the pseudo address for the source of the data is presented to the application. If, in step 714, it is determined that the received packet includes a pseudo address, then it is assumed that there was a change in the connection. In that case, the method loops to step 718 and the table is accessed using the pseudo address in the packet. This pseudo address is used to match the remote pseudo address 404 field of one of the entries in the table. The table entry matching the pseudo address is then updated by replacing the remote global IP address 408 with the IP address from the received packet. After step 720, the method loops to step 716.

Figure 13 is a second embodiment of the process of domain name resolution, which is step 330 of Figure 6. The embodiment described by Figures 13-15 pertain to a scenario where the communication is initiated by entity A (which has a global IP address) with entity B (which is on a private network). In step 740, the application requests domain name resolution, similar to step 350 of Figure 7. In step 742, the domain name is resolved to a global IP address, similar to step 352 of Figure 7. In step 744, entity X uses the global IP address returned in step 742 to contact the gateway. In step 742, the domain name (e.g. B.com) was resolved to the global IP address of the gateway GIP_G. In step 744, GIP_G is used in an IP packet to contact the Gateway and acquire the local address for B, which is LIP_B. In step 746, A chooses a pseudo address to identify B. Step 746 is similar to step 354 of Figure 7. In step 748, the pseudo address chosen for entity B, global address for entity B and local address for entity B

are added to the table. In step 750, the pseudo address for entity B is provided to the application who requested the domain name resolution in step 740.

In one embodiment of step 744, the Gateway stores a table. Each table entry includes at least two fields. One field stores a local address of an entity in the private network and the other field stores the corresponding domain name for the entity associated with the local address. Thus, step 744 includes sending a request to the gateway with a domain name and asking the gateway to return the local address based on the domain name. The domain name could be stored in options field 22 of the IP packet or can be stored within the packet using encapsulation.

Figure 14 is a second embodiment of the process for starting communication, corresponding to step 332 of Figure 6. In step 800, network software 304 on entity A receives a request to communicate from application 302 on entity A. In step 802, entity A creates a packet. The packet includes the global IP address for entity A as the source and the global IP address for the Gateway as the destination. The packet also includes the pseudo address that entity A uses to identify entity B. In step 804, the packet is sent from entity A to the Gateway.

In step 806, the Gateway receives the packet and changes the destination address in the packet from the global IP address for the Gateway (GIP_G) to the local IP address for entity B and sends the edited packet to entity B. In one embodiment, the packet sent from entity A to the Gateway includes the local IP address for entity B. That is, entity A will include the local IP address for entity B in the packet when the packet is created in step 802. In another embodiment, entity A can include the domain name or the pseudo address, which can be used to access a table which associates the domain name or pseudo address with the local addresses. In step 808, entity B receives the packet. In step 810, entity B accesses the pseudo address. In step 812, entity B adds the pseudo address that entity A uses to identify entity B, and entity A's global IP address to the table. In step 814, entity B chooses a pseudo address for entity A and adds that pseudo address to the table in step 816. In step 818, the pseudo address for entity A is provided to application 312 on entity B. In step 820, entity B creates a packet (in response to the application) which includes the global IP address for entity A as the destination and the local IP address for entity B as the source. The packet also includes the pseudo address that entity B uses to identify entity A. In step 822, entity B sends the packet. In step 824, the Gateway receives the packet and changes the source

address from the local IP address of entity B to the global IP address of the Gateway. This packet is then sent to entity A. In another embodiment, entity B can create the IP packet using the global IP address of the Gateway as the source and bypass all or part of step 824. In step 826, entity A receives the packet. In step 828, entity A accesses the pseudo address that entity B used to identify entity A and stores that pseudo address in the table in step 830.

Figure 15 is a second embodiment of the process of communicating, which is step 334 of Figure 6. In step 860, network software 304 of entity A receives data and a pseudo address from application 302 of entity A. The pseudo address is that used by entity A to identify entity B. In step 862, the pseudo address is used to access the table to identify the global IP address for the Gateway (GIP_G) and the local IP address for entity B (LIP_B). In step 864, it is determined whether the connection has changed. Step 864 is similar to step 704 of Figure 12. If the connection has not changed, then in step 866, a packet is created which includes the global IP address of the Gateway as the destination and the global IP address of entity A as the source. The local IP address of entity B is also stored in the packet. In one embodiment, the local IP address is stored in options field 22 of the IP packet. In another embodiment, the local IP address is encapsulated within a packet inside the IP packet. The packet created by step 866 does not include a pseudo address.

If there was a change in the connection (step 864), then in step 870 a packet is created. The packet includes the global IP address of the Gateway as the destination and the global IP address of entity A as the source. Additionally, the pseudo address that entity B uses to identify entity A is added to the packet. After step 866 or step 870, the method proceeds to step 868 and the packet is sent. In step 872, the Gateway receives the packet, replaces the destination of the global IP address for the Gateway with the local IP address of entity B by accessing the local IP address for entity B within the packet and sends the packet to entity B. In another embodiment, the Gateway does not translate the global address of the Gateway to the local address of entity B. Instead, the Gateway encapsulates the incoming packet in a new packet whose destination is the local address of entity B and the source address is the address of the Gateway. This embodiment enables preservation of the packet that originates from the source as is until it reaches the destination, allowing the application of IPsec to the original packet. This enables the use of IPsec end-to-end from source to

destination. IPsec breaks if the content of the original packet is modified along the way. In step 874, entity B receives the packet from the Gateway. If the packet does not include a pseudo address (step 876), then it is assumed that the connection did not change and the method proceeds to step 878. In step 878, the data from the packet is presented to the application along with the pseudo address used by entity B to identify entity A. If, in step 876, it is determined that the packet does contain a pseudo address, then it is assumed that the connection has changed. In step 880, the pseudo address within the packet is used to access the table entry. That table entry accessed in step 880 is updated in step 882 to change the old global IP address of entity A to the new global IP address for entity A found in the current packet. After step 882, the method loops to step 878.

In the above example, the local IP address for entity B is transmitted in Options field 22 of the IP packet. In other embodiments, the system can use encapsulation. That is, multiple IP packets can be encapsulated within each other. The outer IP packet can have a source address of the global IP address of entity A and the destination address of the global IP address for the Gateway. The second IP packet, encapsulated within the first IP packet, has a source equal to the global IP address of the Gateway and a destination address equal to the local IP address of entity B. The third packet, encapsulated inside the second packet, has a source of the global IP address of entity A and destination of the local IP address of entity B. The set of packets is first transmitted from entity A to the Gateway. When the packets are received at the Gateway, the Gateway strips off the first packet and sends the second packet (with the third packet inside) to entity B. When entity B receives the second packet, entity B removes the second packet and uses the third packet to determine how to reply.

In the embodiment described above with respect to Figure 13, the system first performed a regular domain name resolution to determine the global IP address of the Gateway and then contacts the Gateway to get the local IP address of entity B. In one embodiment, entity A can get the local IP address of entity B from the Gateway by doing a domain name resolution request directly with the Gateway. In another embodiment, entity A can obtain a local address for entity B from the Gateway by using an ICMP echo request. In another embodiment, rather than acquiring the global IP address of the Gateway and the local IP address of entity B in two separate steps, the

domain name resolution process can be altered such that when resolving the domain name for entity B (e.g. B.com), the resource record will return with both the global IP address for the Gateway and the local IP address of entity B. In yet another embodiment, the first domain name resolution returns the global address of the Gateway and a second domain name resolution sent to a different entity (other than the Gateway) returns the local address.

In one implementation, the pseudo addresses are communicated using a transport layer header, inserted between the IP header and the TCP or UDP header or a packet. The transport layer header includes the public addresses of the two communicating entities, the pseudo addresses of the two entities and the protocol of the packet payload (TCP or UDP).

The foregoing detailed description of the invention has been presented for purposes of illustration and description. It is not intended to be exhaustive or to limit the invention to the precise form disclosed, and obviously many modifications and variations are possible in light of the above teaching. The described embodiments were chosen in order to best explain the principles of the invention and its practical application to thereby enable others skilled in the art to best utilize the invention in various embodiments and with various modifications as are suited to the particular use contemplated. It is intended that the scope of the invention be defined by the claims appended hereto.

CLAIMS

We Claim:

1. A method for communicating, comprising the steps of:

receiving a request to communicate from a first application on a source, said request includes a first pseudo address corresponding to a destination;

using said first pseudo address to access a public address; and

sending a quantity of information toward said destination using said public address.

2. A method according to claim 1, further comprising the steps of:

receiving said quantity of information at said destination;

providing at least a subset of said quantity of information to a second application on said destination; and

providing a second pseudo address to said second application on said destination, said second pseudo address corresponds to said source.

3. A method according to claim 2, further comprising the steps of:

receiving said quantity of information at an intermediate entity, said quantity of information having a source address and a destination address, said destination address is a public address corresponding to said intermediate entity;

accessing a private address for said destination; and

sending at least a subset of said quantity of information to said destination using said private address for said destination.

4. One or more processor readable storage devices having processor readable code, said processor readable code for programming a processor to perform a method comprising the steps of:

receiving a request to communicate from a first application on a source, said request includes a first pseudo address corresponding to a destination;

using said first pseudo address to access a public address; and

sending a quantity of information toward said destination using said global address.

Fig. 1

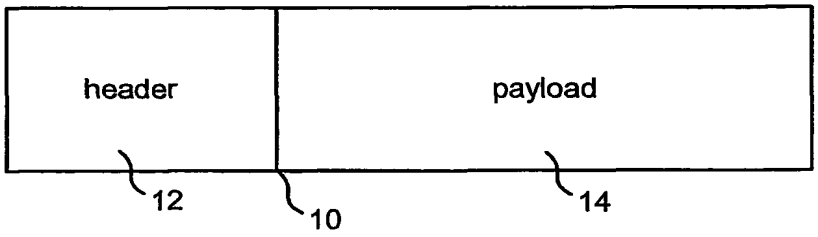


Fig. 2

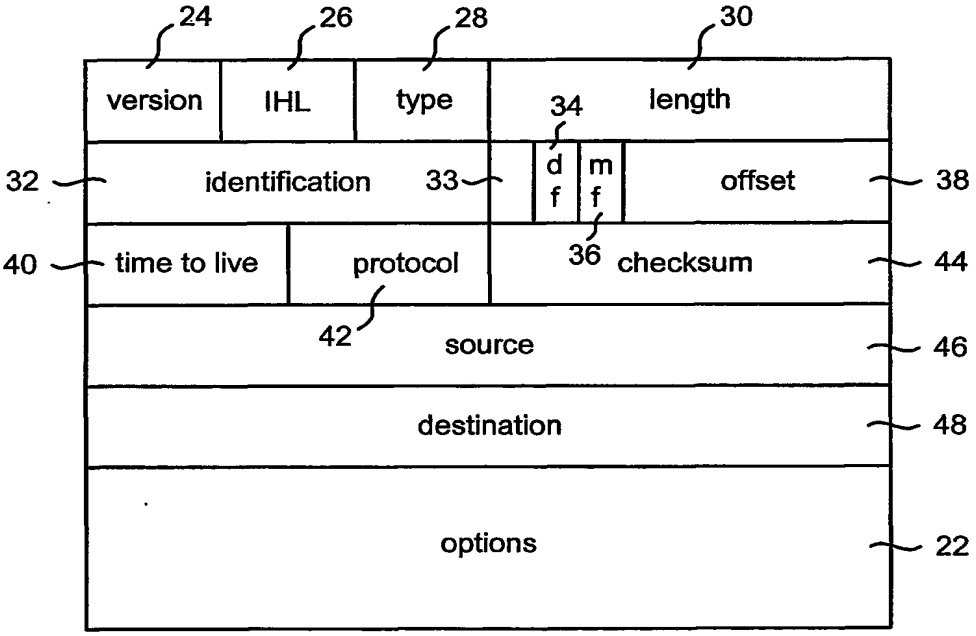


Fig. 8

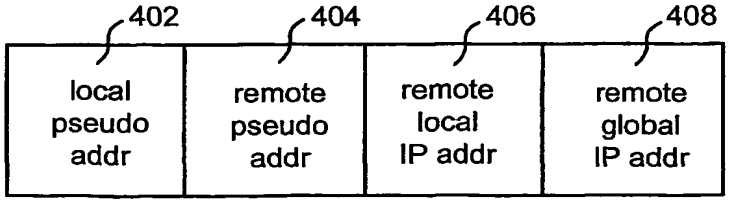


Fig. 3

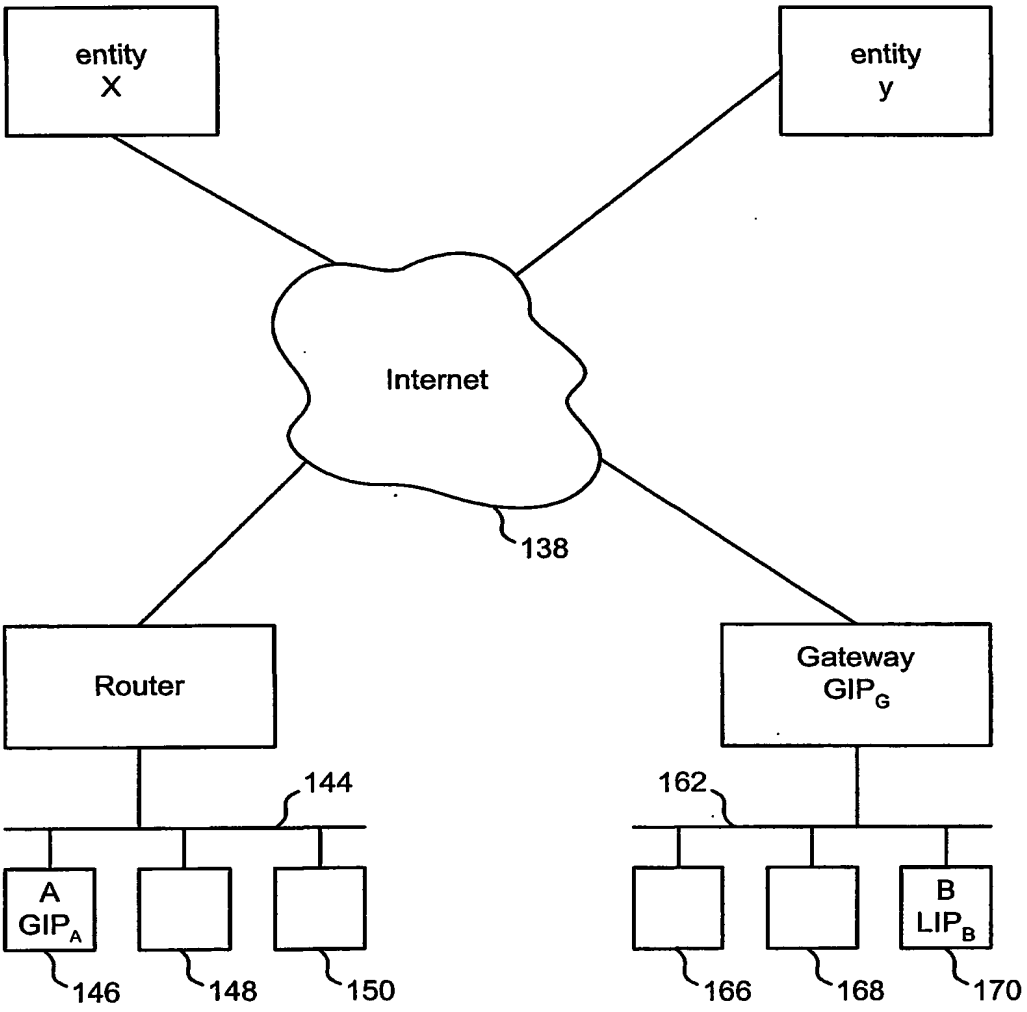

entity X
entity y
Internet
138
Router
Gateway GIP$_G$
144
162
A GIP$_A$
B LIP$_B$
146
148
150
166
168
170

Fig. 4

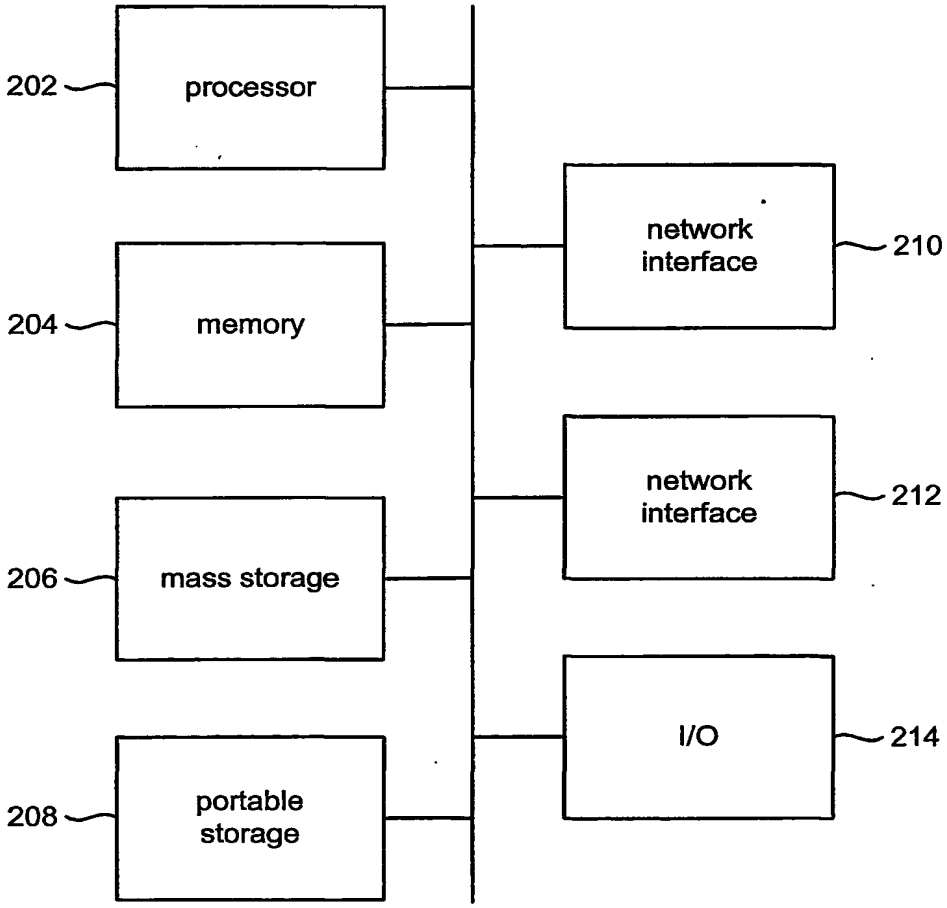

202
processor
204
memory
206
mass storage
208
portable storage
network interface
210
network interface
212
I/O
214

Fig. 5

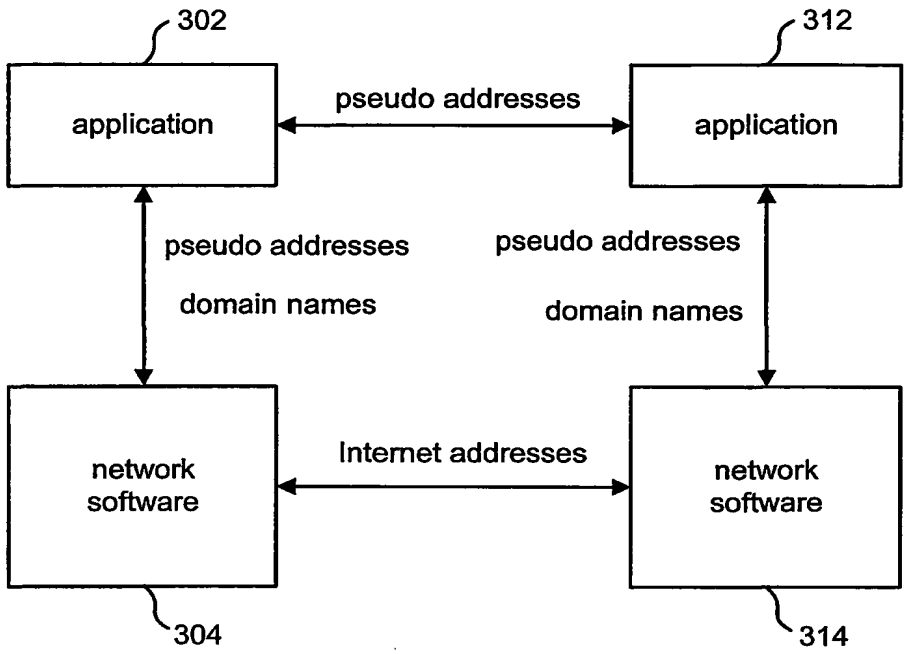

302
312
application
pseudo addresses
application
pseudo addresses
domain names
pseudo addresses
domain names
network software
Internet addresses
network software
304
314

Fig. 10

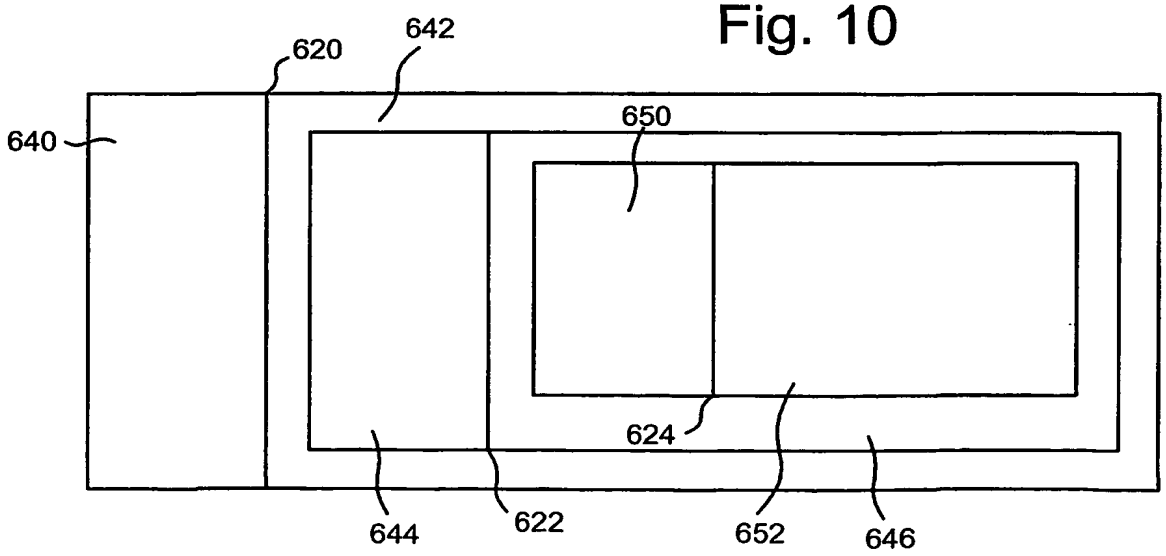

620
642
640
650
624
644
622
652
646

Fig. 6

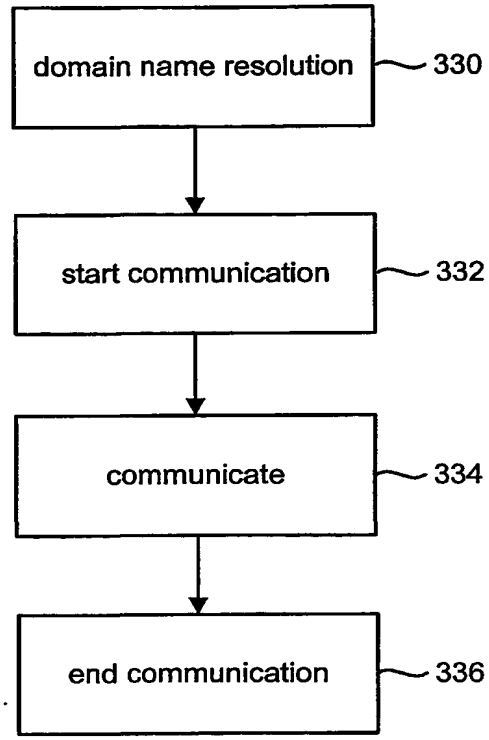


Fig. 7

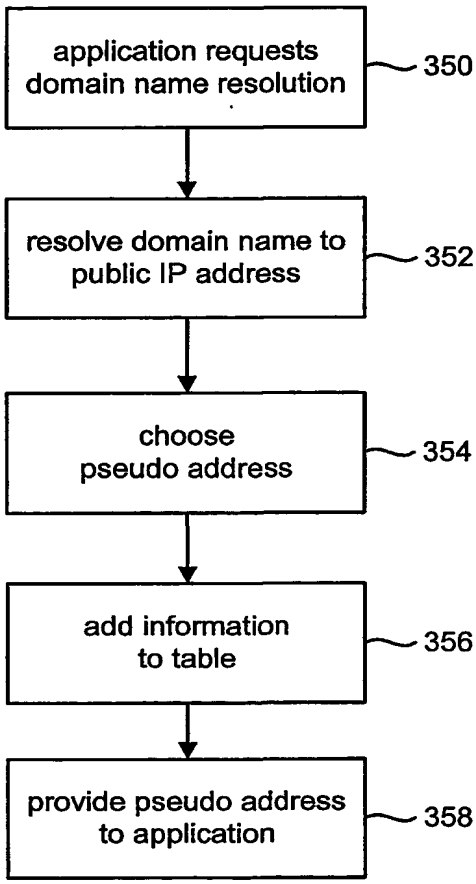


Fig. 11

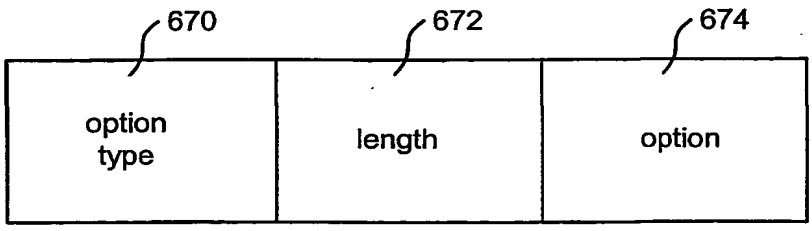
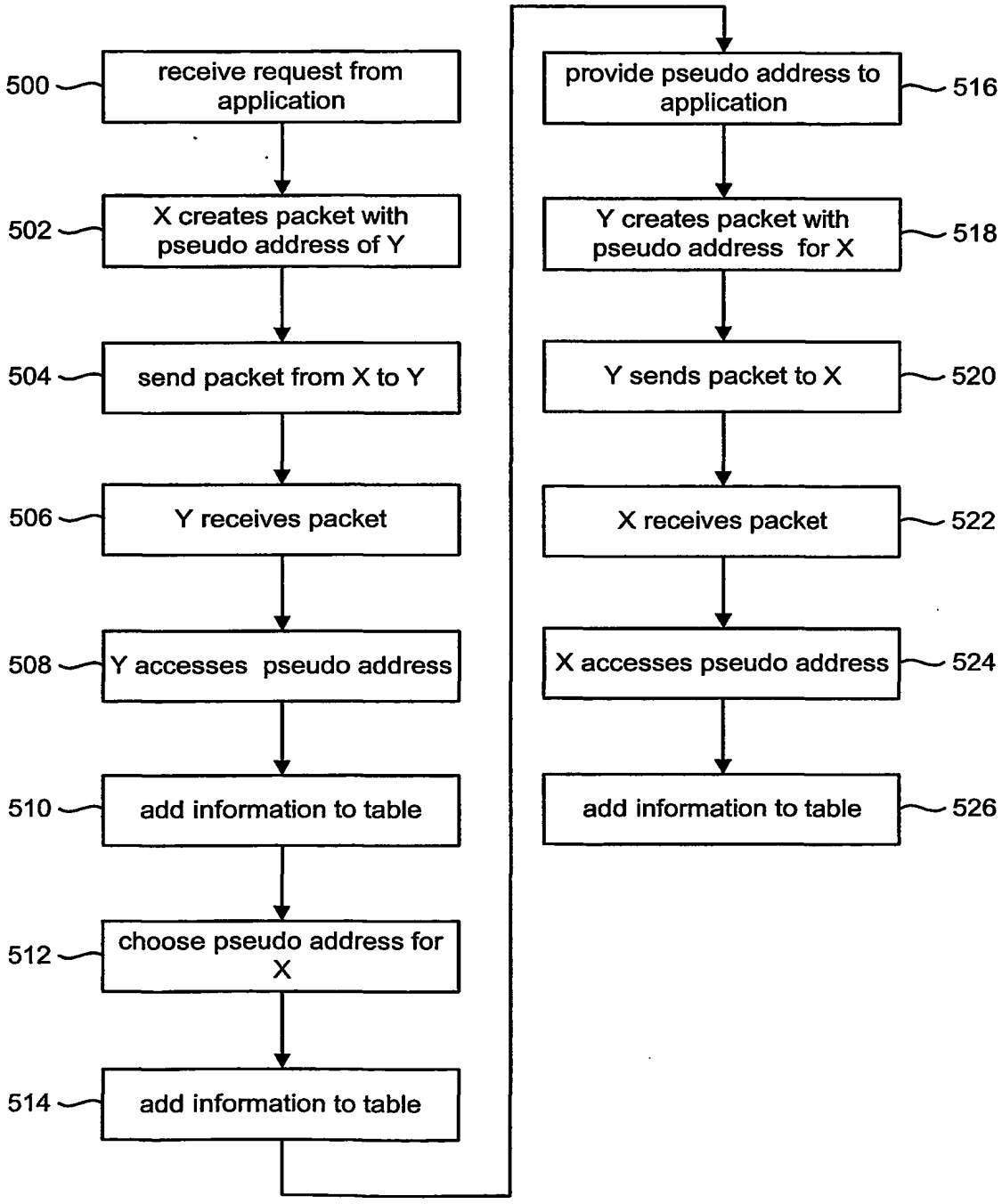


Fig. 9


500
receive request from application
502
X creates packet with pseudo address of Y
504
send packet from X to Y
506
Y receives packet
508
Y accesses pseudo address
510
add information to table
512
choose pseudo address for X
514
add information to table
516
provide pseudo address to application
518
Y creates packet with pseudo address for X
520
Y sends packet to X
522
X receives packet
524
X accesses pseudo address
526
add information to table

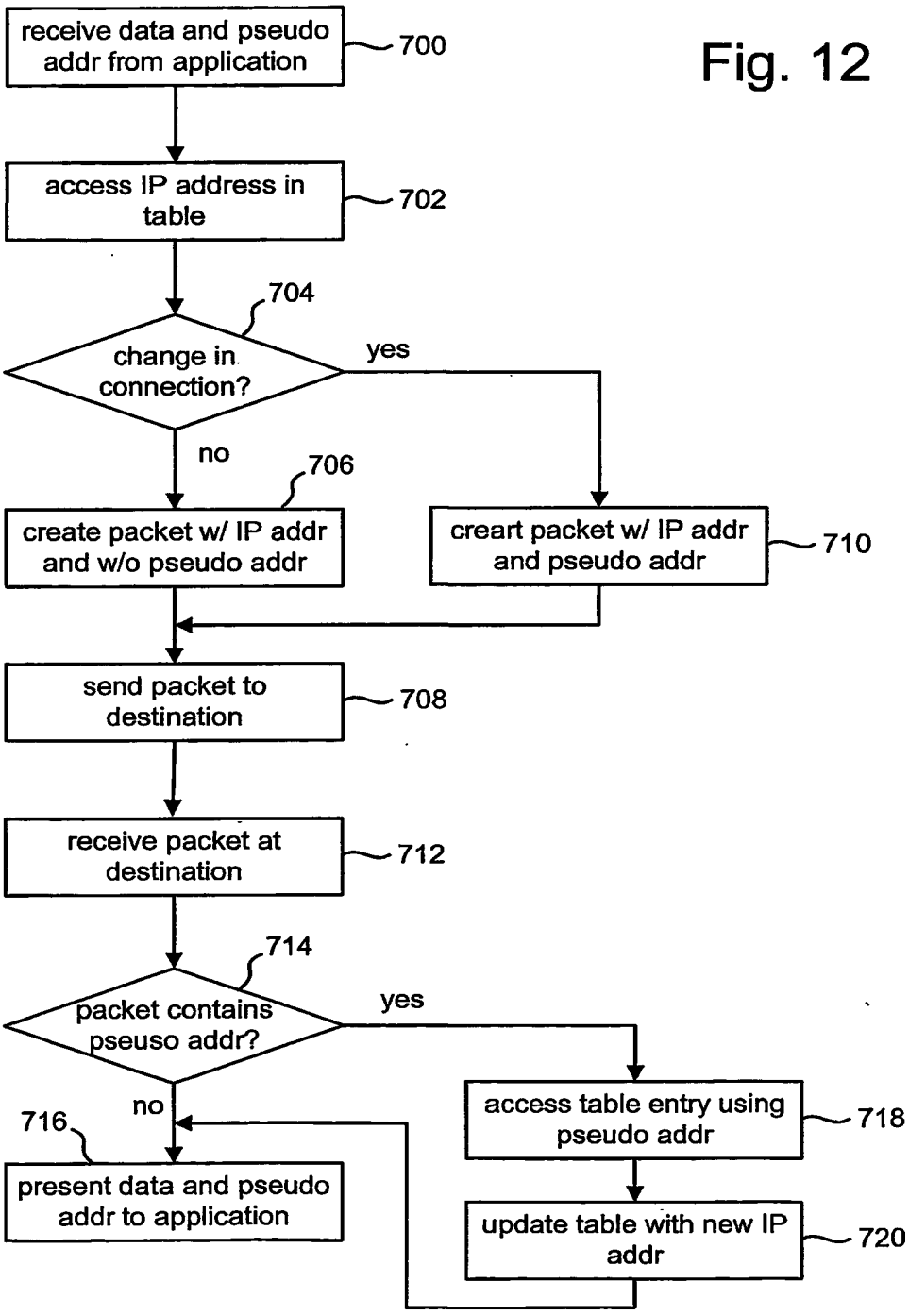

receive data and pseudo addr from application
700
access IP address in table
702
704
change in connection?
yes
no
706
create packet w/ IP addr and w/o pseudo addr
creart packet w/ IP addr and pseudo addr
710
send packet to destination
708
receive packet at destination
712
714
packet contains pseuso addr?
yes
no
716
present data and pseudo addr to application
access table entry using pseudo addr
718
update table with new IP addr
720

Fig. 12

Fig. 13

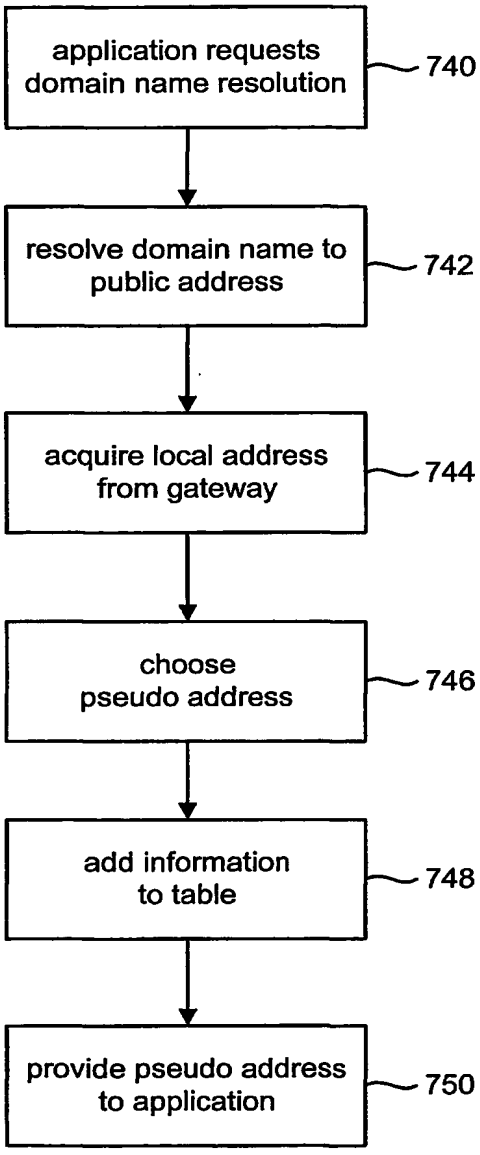
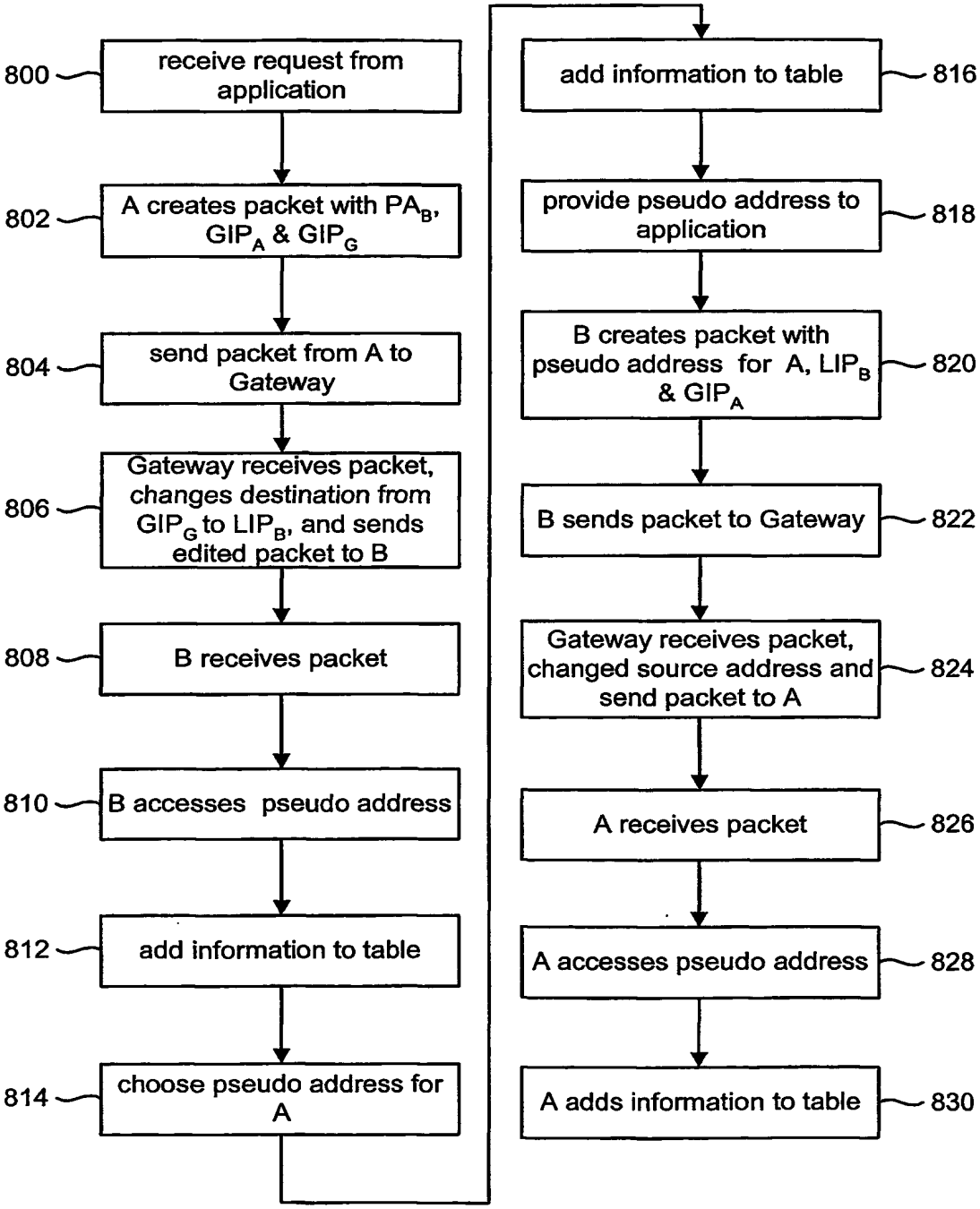

application requests domain name resolution
740
resolve domain name to public address
742
acquire local address from gateway
744
choose pseudo address
746
add information to table
748
provide pseudo address to application
750

Fig. 14


800
receive request from application
802
A creates packet with PA_B, GIP_A & GIP_G
804
send packet from A to Gateway
806
Gateway receives packet, changes destination from GIP_G to LIP_B, and sends edited packet to B
808
B receives packet
810
B accesses pseudo address
812
add information to table
814
choose pseudo address for A
816
add information to table
818
provide pseudo address to application
820
B creates packet with pseudo address for A, LIP_B & GIP_A
822
B sends packet to Gateway
824
Gateway receives packet, changed source address and send packet to A
826
A receives packet
828
A accesses pseudo address
830
A adds information to table

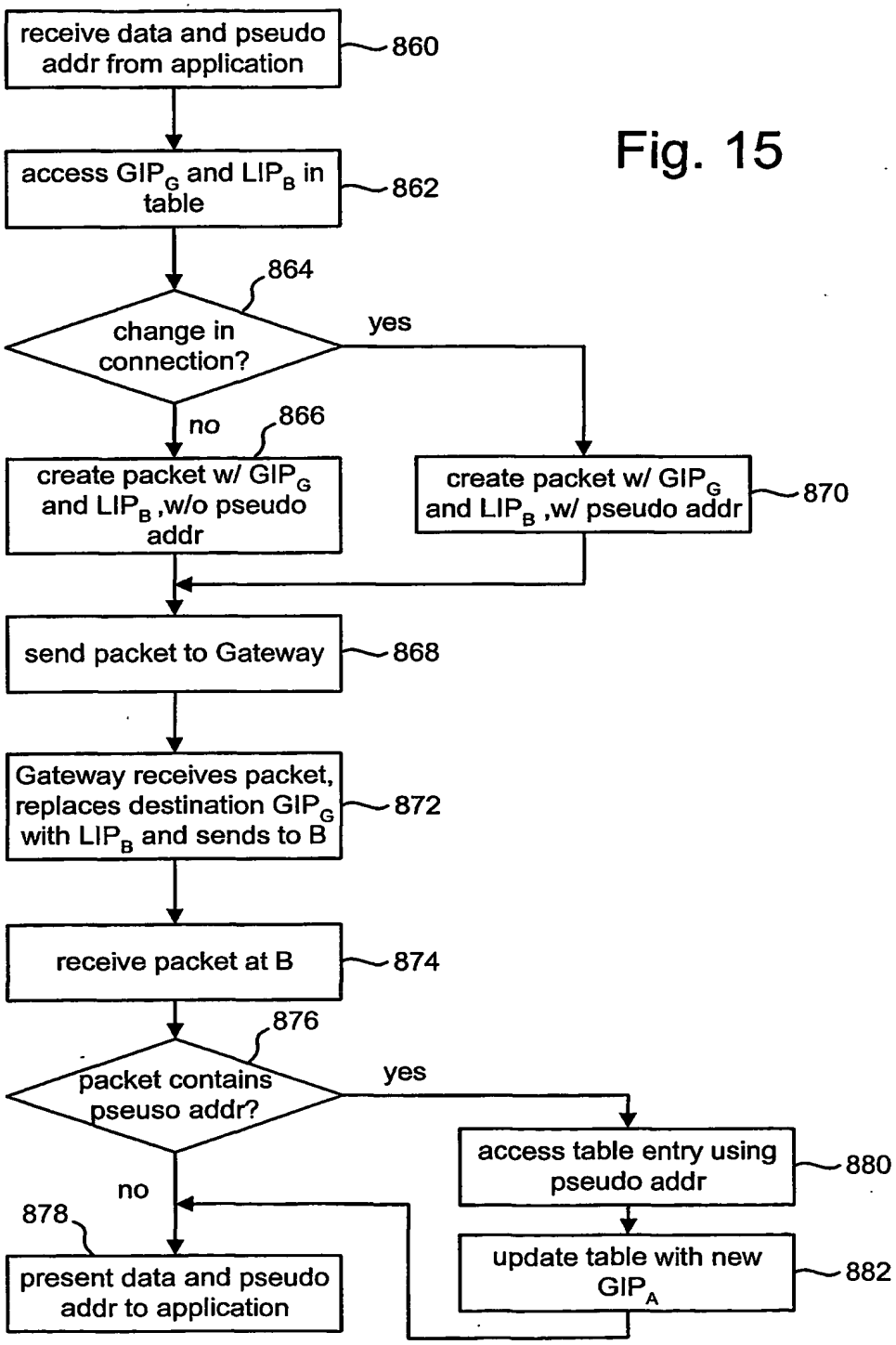

receive data and pseudo addr from application
860
access GIP_G and LIP_B in table
862
864
change in connection?
yes
no
866
create packet w/ GIP_G and LIP_B ,w/o pseudo addr
create packet w/ GIP_G and LIP_B ,w/ pseudo addr
870
send packet to Gateway
868
Gateway receives packet, replaces destination GIP_G with LIP_B and sends to B
872
receive packet at B
874
876
packet contains pseuso addr?
yes
access table entry using pseudo addr
880
no
878
update table with new GIP_A
882
present data and pseudo addr to application

Fig. 15

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US01/23948

A. CLASSIFICATION OF SUBJECT MATTER

IPC(7) :G06F 13/00
US CL : 709/245
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 709/200,217,218,219,230,237,238,245

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

NONE

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

WEST:
search terms: pseudo near1 address$

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 6,101,543 A (ALDEN et al.) 08 AUGUST 2000 see Abstract, figures 1-23, and col. 3 (line 1-et seq.).	1-4
A	US 5,159,592 A (PERKINS) 27 OCTOBER 1992 see Abstract, figures 1-5, col. 2 (line 60-et seq.).	1-4

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

*	Special categories of cited documents:	"T"	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A"	document defining the general state of the art which is not considered to be of particular relevance	"X"	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"E"	earlier document published on or after the international filing date	"Y"	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"L"	document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"&"	document member of the same patent family
"O"	document referring to an oral disclosure, use, exhibition or other means		
"P"	document published prior to the international filing date but later than the priority date claimed		

Date of the actual completion of the international search 13 SEPTEMBER 2001	Date of mailing of the international search report 04 OCT 2001
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230	Authorized officer ROBERT B. HARRELL James R. Matthews Telephone No. (703) 305-9692

Form PCT/ISA/210 (second sheet) (July 1998)*